Jurnal Barekeng
Vol. 8 No. 1 Hal. 7 – 16 (2014)

Barekeng
jurnal ilmu matematika dan terapan
© Jurusan Matematika FMIPA Unpatti 2014

STRUKTUR KOALJABAR UNIVERSAL DALAM SISTEM *STATE-BASED*

Universal CoAlgebra Structures in State-Based System

HENRY W. M. PATTY
Staf Jurusan Matematika Fakultas MIPA Universitas Pattimura
Jl. Ir. M. Putuhena, Kampus Unpatti, Poka-Ambon
E-mail: henrywmpaty81@gmail.com

ABSTRAK

Konsep koaljabar universal yang merupakan dualitas dari aljabar dapat dipandang sebagai suatu teori dalam sistem *state based*. Dalam kotak hitam (*black boxes*), automata dan struktur Kripke yang merupakan contoh sistem *state-based*, struktur koaljabar merupakan penggabungan dua pemetaan yang membawa suatu *state s* ke pasangan elemen dari hasil kali tensor dua himpunan.

Kata kunci : *koaljabar, sistem state-based, kotak hitam, automata, penerima*

PENDAHULUAN

Matematika merupakan struktur formal yang mendasari suatu perhitungan, pengukuran, transformasi, dan lain-lain sebagainya. Dalam matematika khususnya aljabar dikembangkan ide-ide dasar diantaranya sistem bilangan, grup dan ruang vektor dimana struktur dan sifat-sifatnya mendasari ilmu-ilmu yang lain. Misalnya sistem dinamik yang struktur bahasanya adalah aljabar atau yang lebih dikenal sebagai mesin turing. Mesin turing (model komputasi secara teoritis yang ditemukan oleh Alan Mattison Turing pada tahun 1935) merupakan model ideal untuk melakukan perhitungan matematis. Dengan kata lain mesin turing menentukan apakah suatu fungsi dapat diselesaikan dengan komputer atau tidak. Mesin ini masih berupa konsep, sampai kemudian diwujudkan dalam bentuk nyata beberapa tahun kemudian. Secara teori, setiap mesin memiliki *state* yang dapat berubah dari suatu *state* ke *state* yang lain. Dimana para pengguna komputer dapat melihat sebagian *state* lewat layar komputer (printer) dan bahkan dapat merubah *state* komputer dengan meng*input* perintah dan selanjutnya komputer akan menampilkan perilaku *state* tersebut. Namun dalam kenyataannya merupakan suatu sistem yang cukup rumit dideskripsikan secara formal.

Ahli matematika dan computer seperti Rutten [4], Kursz [5] dan Jacobs [6] memperkenalkan beberapa struktur dari sistem *state-based* yaitu automata, sistem transisi, jaring petri (*petri net*), dan struktur-struktur lainnya. Dari beberapa struktur tersebut diabstraksikan konsep koaljabar yang merupakan dual dari aljabar. Hal ini disebabkan karena aljabar universal tidak dapat digunakan dalam memodelkan semua struktur aljabar sehingga dibutuhkan koaljabar universal yang dapat memodelkan sistem state-based [1]. Sistem *state-based* ini dapat dipandang sebagai cikal bakal bahasa pemograman dalam ilmu komputer.

Tipe data dalam bahasa pemrograman komputer tergolong aljabar sedangkan kelas datanya merupakan suatu koaljabar. Suatu tipe data secara lengkap dapat dibedakan lewat pembangunnya, yaitu melalui suatu fungsi aljabar universal dalam bentuk $F(S) \to S$ dengan *state S*. Tipe data ini dapat diabstraksikan dengan fungsi aljabar namun kelas data $S \to F(S)$ tidak dapat diabstraksikan dengan fungsi aljabar, dibutuhkan dualitas dari aljabar yang disebut koaljabar. Untuk mendefinisikan koaljabar ini membutuhkan teori kategori. Misalkan C adalah objek atau kategori dan misalkan $F : C \to C$ adalah fungtor (endofungtor). Selanjutnya *F*-koaljabar atas kategori C didefinisikan sebagai objek S dari C yang dilengkapi dengan morfisma $\gamma_S : S \to F(S)$ dan dinotasikan (S, γ) dimana S merupakan sebarang himpunan dan γ_S merupakan pemetaan.

TINJAUAN PUSTAKA

Suatu koaljabar universal dapat dipandang sebagai teori dari sistem *state-based*. Menurut Hansen & Rutten

[7] koaljabar diperlukan karena (i) dapat mengungkapkan struktur sistem yang kompleks, (ii) sebagai cara pandang baru dari penelitian yang sudah ada sebelumnya, (iii) hasil yang baru atau generalisasi lewat cara pandang secara umum, (iv) dibutuhkan untuk membuktikan teorema teori sistem dan (v) sebagai alat dalam matematika yaitu: morfisma, bisimulasi, ekuivalensi dan model logika.

Sebelum didefinisikan secara formal apa yang dimaksudkan dengan koaljabar, akan diberikan beberapa contoh sistem *state-based* dan morfisma diantara sistem tersebut. Selanjutnya dengan koaljabar serta homomorfisma koaljabar akan diungkapkan sifat-sifat abstrak dan contoh-contohnya.

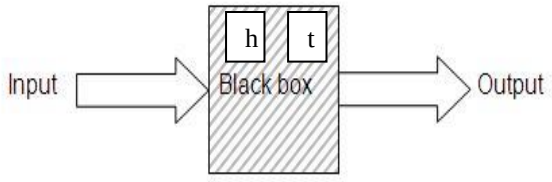
Setiap sistem *state-based* mempunyai *input*, *output* dan *inner state* (keadaan dari dalam sistem tersebut). Perilaku (*bahaviour*) beberapa sistem tidak hanya bergantung dari *input* saja tapi juga dari *inner state* nya dalam pengertian suatu *output* dapat berbeda dengan *input* karena dipegaruhi *inner state* sistem tersebut. Secara umum suatu sistem *state-based* memiliki beberapa karakteristik:(i) perilaku sistem yang tergantung pada *inner state* dimana *state* ini tidak tampak oleh pengguna sistem, (ii) sistem dapat dipengaruhi oleh keadaan disekitarnya, (iii) perilaku sistem berdasarkan operasinya.

Secara khusus, suatu sistem *state-based* membahas tentang perilaku *input-output state* yang diperoleh dan aplikasinya. Suatu sistem juga dapat diterapkan untuk meminimalkan *inner-state* dengan jalan membuang setiap *state* yang tidak diperlukan, sehingga *state* yang ditunjukkan tidak berbeda dengan perilaku *input-output* nya. Pengurangan perbedaan ini dikenal dengan nama bisimulasi (bisimilaritas). Dua *inner-state* s dan s' disebut bisimilar ditulis $s \sim s'$, jika s dan s' tidak dapat dibedakan dari perilaku *input-output* nya dengan anggapan relasi ~ adalah refleksif dan simetris.

Dalam tulisan ini akan diberikan beberapa contoh sistem *state-based* yaitu kotak hitam (*black-boxes*), tipe data (*data stream*), automata dan struktur Kripke dalam sistem yang selanjutnya akan dilihat sebagai koaljabar universal. Sebagian besar teori dalam tulisan ini mengacu pada *Universal Algebra and Coalgebra* yang ditulis Klaus Denecke dan Shelly L. Wismath [1]

1. Kotak-Hitam (Black-Boxes)

Kotak-hitam merupakan suatu kelas khusus dalam sistem *state-based* yang mempunyai layar dan tombol h dan t dimana saat tombol h ditekan maka layar akan menunjukkan suatu elemen data $d \in D$ dengan D=himpunan data. Sedangkan tombol t akan merubah *inner-state* sehingga ketika tombol h ditekan (setelah terlebih dulu di tekan tombol t) maka kotak-hitam akan menampilkan elemen $d' \in D$.



Gambar 1. Kotak Hitam

Misalkan S adalah himpunan *inner-state* dari suatu kotak hitam, maka diperoleh pasangan pemetaan:

$$h : S \to D$$
$$t : S \to S$$

Definisi 1:
Misalkan D adalah himpunan elemen data. Suatu kotak-hitam atas himpunan elemen data D adalah tripel $(S;h,t)$ dimana S adalah himpunan elemen yang disebut state dan $h : S \to D$, $t : S \to S$ adalah pemetaan.

Jika diberikan *inner-state* s dari suatu kotak-hitam, maka akan diperoleh urutan elemen data tak berhingga

$$((h(s),h(t(s)),h(t(t(s))),\ldots)$$

Karena itulah kotak-hitam sering disebut urutan automata (*stream automata*). Selanjutnya mengenai automata akan dijelaskan namun untuk membahas prinsip *black-boxes* sebagai struktur koaljabar perlu dijelaskan tentang hasil kali tensor (*tensor product*) yang dirujuk pada [1]

Definisi 2:
Misalkan $\alpha : D \to E$, $\beta : D \to F$ dimana D, E dan F suatu himpunan data pada sistem state-based. Suatu pemetaan $\alpha \otimes \beta : D \to E \times F$ disebut hasil kali tensor dari α dan β jika dipenuhi $(\alpha \otimes \beta)(d) := (\alpha(d),\beta(d))$, $\forall d \in D$

Definisi 3:
Misalkan $p_1 : A \times B \to A$, $p_2 : A \times B \to B$ adalah pemetaan proyektif pada hasil kali $A \times B$ maka untuk setiap fungsi $h : A \to B$ dan $k : C \to D$ didefinisikan $h \times k : (h \circ p_1) \otimes (k \circ p_2) : A \times C \to B \times D$ yang memenuhi $(h \times k)(a,c) := (h(a),k(c))$ untuk setiap $(a,c) \in A \times C$.

Suatu pemetaan $\alpha \otimes \beta$ memiliki sifat ketunggalan yang merupakan sifat universal dari produk, yaitu jika pemetaan $\alpha : D \to E$, $\beta : D \to F$ maka terdapat suatu pemetaan tunggal yang memenuhi $p_1 \circ (\alpha \otimes \beta) = \alpha$ dan $p_2 \circ (\alpha \otimes \beta) = \beta$.

Dua *state* dalam suatu kotak hitam dapat dibedakan jika dari barisan input yang identik akan memberikan *output* yang berbeda. Oleh karena itu *s~s'* akan berarti $h(s) = h(s')$ dan $t(s) \sim t(s')$, secara ringkasnya disajikan dalam definisi berikut

Definisi 4:
Suatu relasi ~ pada kotak-hitam $(S : h \otimes t)$ adalah relasi $\sim \subseteq S \times S$ yang memenuhi aturan $s \sim s := h(s) = h(s')$ dan $t(s) \sim t(s')$

Suatu bisimulasi terjadi pada kotak hitam $(S : h \otimes t)$ jika setiap relasinya memenuhi Definisi 4 di atas.

Contoh 1.
Misalkan suat kotak hitam dengan delapan elemen yaitu $s_1, \ldots, s_8$. Jika diberikan suatu aksi pada kotak hitam tersebut, berupa suatu pemetaan transisi maka akan diperoleh bahwa setiap *state* s_j akan memiliki *output* $(h(s_j))$ dan anak panah dari s_j ke s_k artinya $t(s_j) = s_k$. Misalkan : $s_1 = s_6 = 33$, $s_2 = s_4 = s_8 = 17$, $s_3 = s_5 = s_7 = 42$

$$(33) \rightarrow (17) \leftrightarrow (42) \leftarrow (17) \leftarrow (42) \leftarrow (33) \quad (42) \leftrightarrow (17)$$

Dari diagram di atas terlihat bahwa dua *state* yang sama dapat dibedakan, misalkan *state* 33. Saat tombol *t* kemudian *h* ditekan akan memberikan angka 17 pada satu sisi dan 42 pada sisi yang lain. Dimulai dari *state* $s_1 = 33$ diperoleh barisan data 33, 17, 42, 17, 42, 33, dari *state* $s_6 = 33$ diperoleh barisan data 33, 42, 17, 42, 17, 33. Terlihat bahwa *state* s_1 dan s_6 dapat dibedakan karena barisan datanya berbeda.
Sedangkan *state* $s_3 = s_5 = s_7 = 42$ akan menghasilkan barisan data yang sama 42, 17, 42, 17, . . .sehingga ketiga *state* ini tidak dapat dibedakan.

2. Urutan Data (*Data Stream*)

Suatu urutan data adalah barisan tak berhingga dimana elemen pertamanya akan disebut kepala (*head*) dan elemen terakhirnya disebut ekor (*tail*). Urutan data dalam kaitannya dengan kotak hitam merupakan barisan tak berhingga dari suatu himpunan data *D* yang dilengkapi dengan pemetaan $\tau : \omega \rightarrow D$ dimana $\omega =$ himpunan terurut bilangan asli dan $\tau(k)$ elemen ke-*k* dari urutan untuk setiap $k \in \omega$

Definisi 5:
Urutan data adalah urutan elemen suatu himpunan D yang dilengkapi dengan dua pemetaan hd dan tl dimana $hd : D^{\omega} \rightarrow D$ *dan* $tl : D^{\omega} \rightarrow D^{\omega}$ *dimana* D^{ω} *diartikan sebagai semua pemetaan dari* ω *ke D maka untuk* $\tau \in D^{\omega}$ *berlaku* $h(\tau) := hd(\tau) := \tau(0)$ dan $t(\tau) := tl(\tau)$ *dimana* $tl(\tau)(k) := \tau(k+1)$

Jika ditulis $(\tau(0), \tau(1), \ldots)$ berarti $hd\ (\tau) = \tau(0)$ adalah kepala dan $tl(\tau) = (\tau(1), \tau(2), \ldots)$ sebagai ekor. Urutan tersebut dapat dipandang sebagai sistem $(D, D^{\omega} : hd \otimes tl)$ dimana $hd \otimes tl : D^{\omega} \rightarrow D \times D^{\omega}$ dengan sifat bahwa setiap dua *state* yang berbeda dapat dibedakan $s \sim s' := s = s$.

3. Automata

Automata adalah mesin abstrak yang dapat mengenali (*recognize*), menerima (*accept*) atau membangkitkan (*generate*) sebuah kalimat dalam bahasa tertentu. Automata berasal dari bahasa Yunani *automatos* yang berarti sesuatu yang bekerja secara otomatis (mesin). Pengertian mesin bukan hanya bersifat elektronis/mekanis saja melainkan segala sesuatu (termasuk perangkat lunak) yang memenuhi ketiga ciri di atas. Aplikasi automata pada perangkat lunak terutama pada pembuatan kompiler bahasa pemrograman komputer. Istilah *automaton* sebagai bentuk tunggal dan automata sebagai bentuk jamak. Teori automata adalah teori tentang mesin abstrak yang : (i) bekerja sekuensial, (ii) menerima input, (iii) mengeluarkan ouput

Automaton tanpa *output* disebut penerima (*acceptor*) atau *recognizer*. Dinotasikan dengan $\mathcal{H} = (I, S; \delta)$ dimana *I=input*, $\varphi \neq S$ =*state*, $\delta : S \times I \rightarrow S$ dengan δ suatu pemetaan transisi. Sedangkan automaton dengan *output* disebut *quintuple* $\mathcal{A} = (I, S, O; \delta, \gamma)$ dimana (I, S, δ) =penerima, *O/D=output*, $\gamma : S \times I \rightarrow O$ =pemetaan *output*. Jadi untuk setiap $s \in S$ dan $e \in I$ nilai $\delta(s, e) = s'$ dan $\gamma(s, e) =$ *output* yang dihasilkan saat *input e* dibaca dan mesin akan menyatakan *s*. Jika semua himpunan *I, S,O* adalah berhingga maka automaton akan berhingga dan sebaliknya . Suatu *I, S,O* yang berhingga dinyatakan dengan I_n, S_n, O_n yang artinya *I, S, O* memuat *n* elemen. Jika δ dan ϕ adalah pemetaan maka hanya ada satu bayangan (*image*) untuk setiap pasangan *state* (*s, e*) maka perilaku automata dapat dibedakan (*deterministic*). Selain itu automata disebut *non-deterministic*.

Ada dua tipe automata yang berbeda bergantung pada *output* dan fungsi γ. Didefinisikan (i) *Mealy* automata jika $\gamma : S \times I \rightarrow O$ adalah fungsi biner dan (ii) *Moore* automata jika $\gamma : S \rightarrow O$ adalah fungsi *unary*. Dalam beberapa kasus *output* tidak bergantung dari elemen *input* saja tapi juga bergantung pada *state*. Jika $s_0 \in I$, dimana s_0 adalah *state* awal dan ditulis

$$(I, S; \delta, s_0) \text{ atau } (I, S, O; \delta, \gamma, s_0).$$

Jika elemennya berhingga, automata dapat disajikan dalam table untuk δ dan γ atau ditampilkan dalam suatu graph berarah. Titik dari graph menunjukkan *state* dan sisi (*edge*) diberi simbol $e_j ; d_k$ dari titik s_2 ke titik s_1 dimana

$$\delta(s_2, e_j) = s_1 \text{ dan } \gamma(s_2, e_j) = d_k.$$

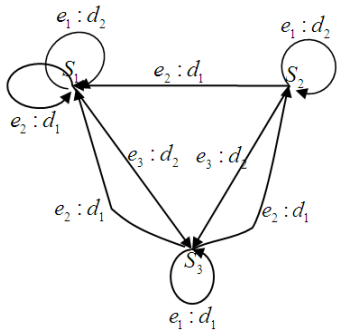
Misalkan

$$S = \{s_1, s_2, s_3\},\ I = \{e_1, e_2, e_3\},\ O = \{d_1, d_2\}$$

ditunjukkan dengan tabel berikut :

δ	e_1	e_2	e_3
s_1	s_1	s_1	s_3
s_2	s_2	s_1	s_3
s_2	s_2	s_1	s_3

γ	e_1	e_2	e_3
s_1	d_2	d_1	d_2
s_2	d_2	d_1	d_2
s_2	d_1	d_1	d_2



Gambar 2. Graph dari Automata

Misalkan s_0 adalah *state* awal dan $e_1, e_2, \ldots, e_n$ adalah barisan *input* maka barisan yang sesuai dengan *state* yang diperoleh adalah $s_0, s_1 = \delta(s_0, e_1), s_2 = \delta(\delta(s_0, e_1), e_2)$ $, \ldots, s_n = \delta(s_{n-1}, e_n)$.

Contoh sebelumnya telah menunjukkan bahwa ada elemen-elemen yang sama dalam barisan dan barisan elemen ouput adalah $d_1 = \gamma(s_0, e_1), \ldots,$ $d_n = \gamma(s_{n-1}, e_n)$. Jelaslah bahwa dua *state* s dan s' dalam automaton tidak dapat dibedakan jika kedua *state* ini menghasilkan *output* yang sama dan *state* yang sama untuk setiap input sehingga didefinisikan suatu bisimulasi untuk automata.

Definisi 6:
Bisimulasi dalam automata adalah relasi $\sim \subseteq S \times S$ *yang memenuhi aturan*
$s \sim s' := (\gamma(s,e) = \gamma(s',e), \delta(s,e) \sim \delta(s',e), \forall s \in I$

Automata yang *finite* (berhingga) digunakan untuk membedakan *bahasa* atau suatu himpunan kata. Dimana bahasa terdiri atas simbol-simbol satuan yang jika dikombinasikan akan mempunyai arti yang berbeda-beda.. Simbol-simbol yang biasa digunakan dalam sebuah bahasa terbatas jumlahnya, yang membentuk sebuah himpunan dan disebut sebagai abjad (*alphabet*). Kadangkala digunakan istilah karakter yang maknanya sama dengan simbol. Deretan karakter membentuk *string*. Bahasa (*language*) didefinisikan sebagai himpunan semua string yang dapat dibentuk dari suatu abjad. Kaidah/aturan pembentukan kata/kalimat disebut tata bahasa (*grammar*). Misalkan terdapat suatu himpunan berhingga $I_n = \{e_1, e_2, \ldots, e_n\}$ merupakan suatu abjad berhingga dengan $n \geq 1$. Dinotasikan $I_n{}^*$ adalah himpunan semesta dari monoid bebas yang dibagun oleh I_n. Setiap bentuk monoid dari $I_n{}^*$ dapat menggambarkan kata yang disusun dari huruf-huruf dalam I_n dengan anggapan bahwa huruf-hurufnya dapat diulang. Contoh e_1e_2, $e_2e_1e_1$ dan $e_1e_2e_3e_3$ adalah kata dari abjad $I_3 = \{e_1, e_2, e_3\}$. Secara umum setiap kata (*word*) dapat diwakili dengan $w = e_{i1}, e_{i2}, \ldots, e_{im}$ untuk suatu $m \geq 0$ dan $e_{i1}, e_{i2}, \ldots, e_{im} \in I_n$. Notasi m adalah panjang kata w dan disimbolkan $|w|$. Jika m=0 berarti merupakan suatu kata kosong (*empty word*) disimbolkan dengan ε. Himpunan $I_n{}^+$ didefinisikan sebagai himpunan semua huruf yang tidak kosong pada abjad I_n. Maka $I_n{}^+$ merupakan suatu semigrup dari monoid $I_n{}^*$ dengan operasi biner.

Diketahui bahasa adalah himpunan kata, secara khusus bahasa atas abjad I_n adalah subset dari semesta monoid $I_n{}^*$. Ada beberapa operasi yang didefinisikan sebagai himpunan bahasa, salah satunya adalah operasi gabungan dalam teori himpunan. Sementara hasil kali dari dua bahasa, katakanlah *U* dan *V* didefinisikan sebagai

$$UV := \{uv \mid u \in U, v \in V\}.$$

Maka akan berlaku

$$U(VW) = (UV)W$$

untuk setiap bahasa *U*, *V*, dan *W* pada I_n, dimana $U\varphi = \varphi U = \varphi$ dan $U\{\varepsilon\} = \{\varepsilon\}U = U$.

Definisi 7:
Hasil kali dari operasi ini dapat diperluas secara induktif menjadi himpunan kuasa dari bahasa (power of language). Untuk setiap bahasa U didefinisikan U^m *untuk semua* $m \geq 0$ *sebagai : (i)* $U^0 = \{\varepsilon\}$ *dan (ii)* $U^m = U^{m-1}U, \forall m \geq 0$

Maka dapat didefinisikan $U^* = \bigcup_{m \in N} U^m$ dan $U^+ = \bigcup_{m \geq 1} U^m$. Suatu kata $w \in I_n{}^*$ akan berada dalam U^* jika dan hanya jika *w* adalah kata kosong, dengan demikian *w* dapat dinyatakan dengan $u_1u_2, \ldots, u_m$ untuk suatu $m \geq 1$. Jadi I_n^m adalah himpunan semua kata dengan dengan panjang *m* pada abjad I_n dan $I_n^* = \bigcup_{m \in N} I_n^m$. Operasi unary yang mengambil bahasa *U* menjadi bahasa U^* disebut iterasi.

Definisi 8:
Suatu himpunan Reg I_n *merupakan himpunan semua bahasa beraturan atas abjad* I_n *adalah himpunan terkecil R sedemikian hingga (i)* $\varphi \in R$ *dan* $\{x\} \in R$, $\forall x \in I_n$ *dan (ii) untuk setiap U dan V dalam R,* $U \cup V$, UV *dan* $U^* \in R$

Dari definisi di atas maka setiap bahasa yang berhingga adalah beraturan. Didefinisikan suatu himpunan *Reg* I_n sebagai himpunan terkecil dari bahasa

atas I_n yang memuat semua bahasa berhingga dan tertutup terhadap tiga operasi bahasa beraturan. Dimana tiga operasi tersebut adalah operasi gabungan, pergandaan dan iterasi. Digunakan simbol biner (+) untuk operasi gabungan, simbol biner ($\circ$) untuk operasi pergandaan dan simbol unary * untuk operasi iterasi. Selain itu dinotasikan operasi nullary untuk ϕ dan kata kosong ε. Jika diberikan tipe aljabar (2, 2, 1, 0, 0), maka tipe ini disebut ekspresi regular atas abjad I_n.

Definisi 9:
Suatu automaton atau acceptor $\mathcal{A}=(I,S,O;\delta,\gamma)$, *dengan pemetaan output unary* $\gamma: S \to O$ *,dan pemetaan transisi* $\delta^*: S \times I_n^* \to S$ *akan memenuhi :*

(i) $\delta^*(s,\varepsilon)=s,\ \forall s \in S$ *dan*

(ii) $\delta^*(s,ew)=\delta^*(\delta(s,e),w),\ \forall w \in I^*, e \in I_n$.

Maka bisimulasi dari automata dapat didefinisikan dengan aturan sebagai berikut: $s \sim s' := \gamma(s)=\gamma(s'),\ \forall e \in I_n$ $(\delta(s,e) \sim \delta(s',e))$

Diperkenalkan suatu kongruensi baru yang didefinisikan pada setiap automata, dimana akan dibahas karakteristik dari bisimulasi yang terbesar pada automaton.

Definisi 10:
Misalkan $\mathcal{A}=(I,S,O;\delta,\gamma)$ *adalah automaton. Suatu Nerode atau kongruensi sintatic pada* $\mathcal{A}$, *adalah relasi* $\sim_N$ *yang didefinisikan sebagai berikut:*

$$(\forall s,s' \in S)\ s \sim_N s' \Leftrightarrow \forall w \in I^*, (\gamma(\delta^*(s,w)) = \gamma(\delta^*(s',w))$$
$$\Leftrightarrow \forall w \in I^*\ ((\delta^*(s,w),\delta^*(s',w)) \in Ker\,\gamma$$
$$\Leftrightarrow \forall w \in I^*\ ((s,w),(s',w)) \in Ker(\gamma \circ \delta^*)$$

Proposisi 1:
Misalkan $\mathcal{A}=(I,S,O;\delta,\gamma)$ *adalah suatu many-sorted aljabar yang mewakili suatu automata. Kongruensi Nerode adalah bisimulasi terbesar dan kongruensi terbesar* φ *pada* $\mathcal{A}$ *yang memenuhi* $\varphi \subseteq Ker\,\gamma$

Bukti

Jelas berdasarkan definisi $\sim_N$ adalah relasi ekuivalensi pada S. Untuk melihat bahwa $\sim_N$ adalah relasi kongruensi, dimisalkan *state* s dan s' maka $s \sim_N s'$, dan $\forall e \in I, w \in I^*$ diperoleh :

$$\gamma(s) = \gamma(\delta^*(s,\varepsilon)) = \gamma(\delta^*(s',\varepsilon)) = \gamma(s')$$

Maka $\forall e \in I, w \in I^*$ diperoleh :

$$\gamma(\delta^*(\delta(s,e),w)) = \gamma(\delta^*(s,ew) = \gamma(\delta^*(\delta(s',e),w))$$

$$\therefore \delta(s,e) \sim_N \delta(s',e)$$

Misalkan ~ bisimulasi pada S, akan ditunjukkan $\sim \subseteq \sim_N$

Akan dibuktikan bahwa untuk setiap kata $w \in I^*$ dan untuk semua *state* $s, s' \in S$ berlaku

$$s \sim s' \Rightarrow \gamma(\delta^*(s,w)) = \gamma(\delta^*(s',w))$$

Dengan induksi :

Untuk $w=\varepsilon$, jelas dipenuhi karena langsung dari definisi bisimulasi

Untuk $w=ev$, untuk suatu huruf e dan untuk kata v dimana :

$$\gamma(\delta^*(s,v)) = \gamma(\delta^*(s',v)) \Rightarrow$$
$$\gamma(\delta^*(s,ev)) = \gamma(\delta^*(\delta(s,e),v)) = \gamma(\delta^*(\delta(s',e),v)) = \gamma(\delta^*(s',ev))$$

$$\therefore \sim \subseteq \sim_N$$

Untuk setiap bisimulasi ~ dipunyai $\sim \subseteq Ker\,\gamma$, maka akan ditunjukkan bahwa $\theta \subseteq Ker\,\gamma$ adalah kongruensi dari *many-sorted* aljabar $A=(I,S,O;\delta,\gamma)$

Misalkan (s,s') adalah pasangan dalam θ. Jika θ adalah kongruensi maka $\gamma(s)=\gamma(s')$ dan $\delta^*(s,\varepsilon) = \delta^*(s',\varepsilon)$. Anggaplah $w=eu$ untuk suatu huruf e dan kata u dimana $\delta^*(s,u)=\delta^*(s',u)$ maka

$$(\delta(s,e),\delta(s',e)) \in \theta$$

dan

$$\delta^*(\delta(s,e),u) = \delta^*(\delta(s',e),u)$$

diperoleh :

$$\delta^*(s,eu) = \delta^*(\delta(s,e),u) = \delta^*(\delta(s',e),u) = \delta^*(s',eu)$$

$$\therefore \theta \subseteq \sim_N \quad \square$$

Diberikan suatu himpunan *state S* sebagai himpunan automata berhingga yang deterministic, dengan *state* s_0 adalah *state* awal dan himpunan $F \subseteq S$ adalah akhir (*final*)/himpunan *state* yang diterima. Suatu kata (*word*) dikatakan dapat diterima oleh automaton jika $\delta^*(s_0,w) \in F$. Misalkan $\mathcal{L}(\mathcal{A})$ adalah himpunan bahasa yang dapat diterima oleh $\mathcal{A}$. Suatu bahasa $\mathcal{L}$ dikatakan dapat dikenali jika terdapat automata berhingga yang *deterministic* atau penerima $\mathcal{A}$ sedemikian hingga $\mathcal{L}=\mathcal{L}(\mathcal{A})$. Teorema Kleene mengatakan bahwa suatu bahasa dapat dikenali jika dan hanya jika bahasa tersebut regular.

Definisi 11:
Misalkan L adalah bahasa atas abjad I maka $\forall e \in I$ *didefinisikan*:

$$L_e := \{ w \in I^* \mid ew \in L \} \qquad = \gamma(\delta^*(s',ew)$$

Dapat diterapkan setiap state s dari automaton (penerima) A yang memuat semua kata yang berasal dari s ke dalam suatu state yang diterima

$$L(A,s) := \left\{ w \in I^* \middle| \delta^*(s,w) \in F \right\}$$

Proposisi 2:
Misalkan S adalah himpunan state dari suatu automaton A atas abjad I dan misalkan $F \subseteq S$ adalah state akhir dari A. $(\forall s,s' \in S)(\forall e \in I)$ berlaku :

$$\delta(s,e) = s' \Leftrightarrow L(A,s)_e = L(A,s')$$

Bukti

$(\Rightarrow)$ Diketahui : $\delta(s,e) = s'$

Akan dibuktikan : $L(A,s)_e = L(A,s')$

$$\begin{aligned} L(A,s)_e &= \left\{ w \in I^* \middle| ew \in L(A,s) \right\} \\ &= \left\{ w \in I^* \middle| \delta^*(s,ew) \in F \right\} \\ &= \left\{ w \in I^* \middle| \delta^*(\delta(s,e),w) \in F \right\} \\ &= \left\{ w \in I^* \middle| \delta^*(s',w) \in F \right\} \\ &= L(A,s') \end{aligned}$$

$(\Leftarrow)$ Diketahui : $L(A,s)_e = L(A,s')$

Akan dibuktikan : $\delta(s,e) = s'$

$$\begin{aligned} L(A,s)_e = L(A,s) &\Leftrightarrow \left\{ w \in I^* \middle| \delta^*(\delta(s,e),w) \in F \right\} \\ &= \left\{ w \in I^* \middle| \delta^*(s',w) \in F \right\} \end{aligned}$$

$\therefore \delta(s,e) = s'$ □

4. Struktur Kripke

Banyak informasi dalam sistem *state-based* yang terdiri dari beberapa komponen berupa kumpulan program yang berhubungan satu sama lain dan saling bekerja sama. Namun beberapa operasi/interaksinya tidak dapat dibedakan dan bahkan dibutuhkan suatu pemetaan untuk menentukan *state* yang berubah dan *ouputnya* sebagai himpunan batasan. Batasan yang diberikan berupa suatu relasi *R* tidak tunggal dari sisi kanan. Suatu relasi *R* yang tidak tunggal dari sisi kanan berupa relasi antara pasangan (a,b) dan (a,c) dengan $b \neq c$. Struktur yang yang dimodelkan lewat automata yang tidak dapat dibedakan tersebut dikenal dalam ilmu komputer sebagai struktur Kripke. Suatu sifat utama dari struktur Kripke adalah relasi transisi dan bukan fungsi transisi seperti pada automata.

Definisi 12:
Misalkan Φ suatu himpunan tak kosong. Struktur Kripke atas Φ adalah tripel $(S;R;\upsilon)$ dengan S himpunan state, R relasi biner dalam S ($R \subseteq S \times S$) dan fungsi $\upsilon:\Phi \to P(S)$ dimana $P(S)$ adalah himpunan kuasa dari S. Selanjutnya pasangan $(S;R)$ disebut bingkai Kripke.

Suatu perubahan dari *state s* ke *state* s' dalam suatu struktur Kripke dinyatakan dalam bentuk pasangan dalam relasi *R*. Hal ini disebut relasi transisi (dinotasikan $s \xrightarrow{R} s'$). Untuk menyatakan struktur Kripke dalam suatu operasi tunggal pada *S* dapat dikombinasikan relasi transisi $R \subseteq S \times S$ dan suatu fungsi valuasi $\upsilon:\Phi \to P(S)$ ke dalam suatu pemetaan dengan domain *S*. Langkah pertama, dapat dibawa informasi dalam relasi *R* ke dalam bentuk pemetaan.

Misalkan $R \subseteq A \times B$ adalah sebarang relasi. Selanjutnya didefinisikan suatu pemetaan parsial $f:A \to P(B)$ dengan $x \sim y \wedge x \to x' := (\exists y')y \to y' \wedge x' \sim y'$ dimana $C := \left\{ c \in B \middle| (a,c) \in R \right\}$ hal ini berarti setiap elemen *a* dalam himpunan semua elemen saling berelasi dengan relasi *R*. Sebaliknya setiap pemetaan $f:A \to P(B)$ dalam relasi $R_f : A \times B$ sebagai $R_f := \left\{ (a,c) \middle| c \in C \ni f(a) = C \right\}$. Hal ini berakibat ada korespondensi satu-satu antara relasi $R \subseteq A \times B$ dan pemetaan parsial $f:A \to P(B)$.

Dalam kaitannya dengan struktur Kripke dapat dibawa (diubah) relasi transisi $R \subseteq S \times S$ ke dalam suatu fungsi induksi $next: S \to P(S)$. Selanjutnya dengan penambahan suatu pemetaan $prop: S \to P(\Phi)$ yang didefinisikan sebagai $s \to \left\{ a \in \Phi \middle| s \in \upsilon(a) \right\}$. Karena kedua pemetaan tersebut memiliki domain yang sama *S* maka dapat digunakan hasil kali tensor $next \otimes prop: S \to P(S) \times P(\Phi)$. Sehingga struktur Kripke dari $(S;R;\upsilon)$ dapat dinyatakan sebagai pasangan $(S;next \otimes prop)$.

Untuk mendefinisikan bisimulasi dalam struktur Kripke dibutuhkan dua *state*. Untuk dua state yang dapat dibedakan katakanlah *x* dan *y*, bisimulasinya adalah $x \sim y := \upsilon(x) = \upsilon(y)$. Hal ini berarti kedua *state* tersebut menyatakan dua proposisi yang berbeda dan sahih (*well defined*). Sedngkan untuk dua *state* yang tidak dapat dibedakan,harus dikondisikan bahwa untuk setiap transisi yang dimulai dari *x* harus ada transisi yang dimulai dari *y*. Didefinisikan bisimulasi dari *x* dan *y* sebagai berikut:

$$x \sim y \wedge x \to x' := (\exists y')y \to y' \wedge x' \sim y'$$

dan

$$x \sim y \wedge y \to y' := (\exists x')x \to x' \ \wedge x' \sim y'.$$

HASIL DAN PEMBAHASAN

Dalam bagian ini akan dibahas prinsip-prinsip koaljabar dalam sistem *state-based* . Konsep koaljabar dalam sistem *state-based* dapat dilihat dari beberapa contoh berikut ini.

a. Kotak Hitam (*Black boxes*)
Kotak-hitam dapat dipandang sebagai suatu struktur aljabar. Untuk suatu himpunan data *D* dan himpunan *state S* yang dilengkapi dengan pemetaan

$$h:S \to D,\ t:S \to S$$

maka dapat dibentuk $(S,D;h,t)$. Selanjutnya kotak hitam dapat dipandang juga sebagai suatu koaljabar

dengan menggabungkan dua pemetaan h dan t ke dalam satu pemetaan $\alpha: S \to N \times S$ dengan definisi

$$\alpha(s) := (h \otimes t)(s) . = (a, s')$$

b. Rekening Bank (*Bank Account*)
Dalam sistem program rekening bank, ada dua pemetaan yang bisa dibentuk
$show: S \to Z;\ trans: S \times Z \to S$
dimana keduanya dapat digabungkan menjadi suatu pemetaan

$$\alpha: S \to Z \times S^Z .$$

Pemetaan ini membawa setiap *state* $s \in S$ menjadi pasangan: bilangan bulat dan suatu pemetaan dari Z ke S

c. Automaton
Untuk suatu automata berhingga (tipe Moore Automata) dua pemetaan

$$\gamma : S \to D,\ \delta : S \times I \to S$$

dapat digabungkan menjadi suatu pemetaan

$$\alpha: S \to D \times S^I .$$

Pemetaan α memetakan setiap *state* s ke pasangan: elemen *output* dan pemetaan dari setiap himpunan *input* I ke *state* S.

d. Penerima (*Acceptor*)
Untuk sebuah penerima dengan himpunan $F \subseteq S$, dimana F adalah *state* akhir (*final state*) dapat digabungkan dua pemetaan : F dan pemetaan

$$\delta: S \times I \to S\ ,\ \alpha: S \to \{0,1\} \times S^I .$$

Pemetaan α membawa setiap *state* s ke pasangan: $\{0,1\}$ (dimana 0:=tidak diterima, 1:=diterima) dan pemetaan dari I ke S.

e. Struktur Kripke
Telah dijelaskan sebelumnya dalam suatu struktur Kripke $(S; R; \upsilon)$, relasi $R \subseteq S \times S$ dapat digantikan dengan suatu pemetaan

$$next: S \to P(S)$$

Untuk memperoleh setiap pemetaan *next* dianggap bahwa setiap relasi R menggunakan setiap state S seabagai komponen pertama. Selanjutnya dengan kombinasi pemetaan
$prop: S \to P(\Phi)$
dapat dipandang struktur Kripke $(S; R; \upsilon)$ sebagai $(S; next \otimes prop)$
dimana
$next \otimes prop: S \to P(S) \times P(\Phi)$

Dari beberapa contoh yang telah dikemukakan di atas diperoleh suatu struktur koaljabar $(A; \alpha_A)$ dimana $\alpha_A : A \to F(A)$ merupakan suatu fungsi dari sebarang himpunan A ke suatu fungtor $F(A)$. Hal ini dapat didefinisikan sebagai berikut.

Definisi 12:
Suatu F-koaljabar atau biasa disingkat koaljabar adalah suatu sistem $(A; \alpha_A)$ *memuat himpunan A dan pemetaan* $\alpha_A : A \to F(A)$ *untuk konstruksi teori himpunan F(A).*

Homomorfisma Sistem *State-based*

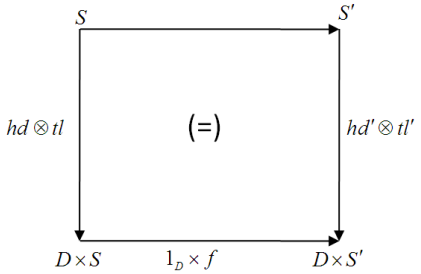
Definisi 13:
Misalkan (S, hd, tl) *dan* (S', hd', tl') *adalah kotak hitam. Suatu pemetaan* $f : S \to S'$ *disebut homomorfisma dari kotak hitam jika untuk setiap* $s \in S$ *berlaku :*
(i) $hd(s) = (hd' \circ f)(s) = hd'(f(s))$ *dan*
(ii) $f(tl(s)) = (tl' \circ f)(s) = tl'(f(s))$
Dalam hal ini yang dimaksudkan dengan representasi koaljabar dari suatu kotak hitam, yaitu jika kondisi (i) dan (ii) terpenuhi. Maka untuk setiap $s \in S$ berlaku :

$$(hd(s), (f \circ tl(s)) = ((hd' \circ f)(s), (tl' \circ f)(s))$$

atau dapat ditulis dalam bentuk

$$(1_D \times f) \circ (hd \otimes tl) = (hd' \otimes tl') \circ f \qquad (*)$$

Dimana $1_D : D \to D$ adalah pemetaan identitas pada D. Persamaan (*) dapat dilihat dari diagram komutatif berikut ini :



Gambar 3. Homomorfisma kotak hitam

Proposisi 3:
Suatu pemeteaan $f : S \to S'$ *adalah homomorfisma dari kotak hitam jika dan hanya jika* $(1_D \times f) \circ (hd \otimes tl) = (hd' \otimes tl') \circ f$

Bukti
($\Rightarrow$) Jelas berdasarkan definisi homomorfisma suatu kotak hitam/setiap homomorfisma f dari kotak hitam akan memenuhi kondisi (*)
($\Leftarrow$) Dengan menggunakan sifat umum dari suatu produk kartesian bahwa untuk setiap pemetaan $f : C \to A$ dan $g : C \to B$ maka terdapat suatu pemetaan tunggal $f \otimes g : C \to A \times B$ yang memenuhi: $p_1 \circ (f \otimes g) = f$ dan $p_2 \circ (f \otimes g) = g$ dimana p_1 dan p_2 adalah suatu pemetaan proyeksi/kanonik.

Anggaplah kondisi (*) terpenuhi maka dengan suatu pemetaan f maka untuk $\forall s \in S$ berlaku :
Dari ruas kiri diperoleh :

$$(1_D \times f) \circ (hd \otimes tl)(s) = (1_D \times f)(hd(s), tl(s))$$
$$= (1_D \circ hd)(s), (f \circ tl)(s))$$
$$= (1_D(hd(s)), f(tl(s))$$
$$= (hd(s), f(tl(s)))$$

Dari ruas kanan diperoleh :

$$((hd' \otimes tl') \circ f)(s) = hd'(f(s)) \otimes tl'(f(s))$$
$$= (hd'(f(s)), tl'(f(s))$$

Jadi kondisi (*) terpenuhi jika dan hanya jika $\forall s \in S$ diperoleh $(hd(s), f(tl(s))) = (hd'(f(s)), tl'(f(s)))$. Dengan kata lain $hd(s) = hd'(f(s))$ dan $f(tl(s)) = tl'(f(s))$ yang tidak lain merupakan definisi dari homomorfisma suatu kotak hitam. □

Secara umum untuk suatu operasi $f : A \to A$ himpunan kuasa f^n secara induktif didefinisikan sebagai $f^0 = id_A$, $f^n = f \circ f^{n-1}$ untuk $n > 0$. Setiap kotak hitam dapat menghasilkan suatu barisan $(hd(s), (hd \circ tl)(s), (hd \circ tl^2)(s), (hd \circ tl^3)(s), \ldots)$ $\forall s \in S$

Proposisi 4:
Misalkan $(S; hd, tl)$ *dan* $(S'; hd', tl')$ *adalah kotak hitam dan* $f : S \to S'$ *adalah suatu homomorfisma, maka untuk setiap* $s \in S$ *berlaku* $(hd \circ tl^n)(s) = (hd' \circ (tl')^n)(f(s))$

Bukti
Akan ditunjukkan dengan suatu induksi pada n bahwa $f \circ tl^n = (tl')^n \circ f$ dimana $f : S \to S'$ adalah homomorfisma suatu kotak hitam.
Untuk $n = 0$ jelas berlaku. Untuk $n = 1$ memenuhi kondisi (ii) homomorfisma kotak hitam
Untuk $n > 1$ dan misalkan $s \in S$ maka diperoleh :
Dari ruas kiri diperoleh :

$$(f \circ tl^n)(s) = (f \circ tl \circ tl^{n-1})(s)$$
$$= (tl' \circ f \circ tl^{n-1})(s)$$
$$= (tl' \circ (tl')^{n-1} \circ f)(s)$$
$$= ((tl')^n \circ f)(s)$$

Dari ruas kanan diperoleh :

$$(hd' \circ (tl')^n)(f(s)) = (hd' \circ (tl')^n \circ f)(s)$$
$$= (hd' \circ f \circ tl^n)(s)$$
$$= (hd \circ tl^n)(s) \quad \square$$

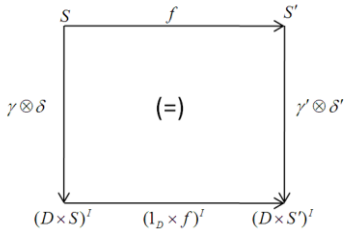
Suatu Automata yang ditetapkan dengan *input* abjad I, *output* abjad D maka dapat dibentuk suatu automata berhingga yang deterministik berupa triple $(S; \delta, \gamma)$ dimana
(i) $\delta : S \times I \to S$ adalah pemetaan transisi
(ii) $\gamma : S \times I \to D$ adalah pemetaan ouput.

Definisi 14:
Misalkan $(S; \delta, \gamma)$ *dan* $(S'; \delta', \gamma')$ *adalah automata. Suatu pemetaan* $f : S \to S'$ *adalah homomorfisma automata jika untuk setiap* $s \in S$ *dan setiap* $e \in I$ *berlaku: (i)* $\gamma(s, e) = \gamma'(f(s), e)$ *dan (ii)* $(f \circ \delta)(s, e) = \delta'(f(s), e)$

Kondisi (i) dari definisi di atas mempunyai arti bahwa *state* s dan $f(s)$ mempunyai *output* yang sama untuk setiap *input* e. Jika $\gamma(s, e) \in D$ dan $f : S \to S'$ maka fungsi f terjadi hanya pada satu bagian saja, hal ini membuat definisi fungsi f berbeda dengan definisi homomorfisma secara umum.
Kondisi (ii) dari definisi di atas mempunyai arti bahwa f kompatibel dengan pemetaan δ dan δ'

Misalkan $f : A \to B$ adalah pemetaan dan misalkan C adalah sebarang himpunan, maka A^C menotasikan setiap pemetaan dari C ke A. $f^C : A^C \to B^C$ adalah pemetaan yang membawa setiap $h \in A^C$, sehingga $f^C(h) := f \circ h \in B^C$

Proposisi 5:
Misalkan $(S; \delta, \gamma)$ *dan* $(S'; \delta', \gamma')$ *adalah automata berhingga yang deterministik. Suatu pemetaan* $f : S \to S'$ *adalah homomorfisma automata jika dan hanya jika diagram di bawah ini komutatif*



Gambar.4. Homomorfisma automata

Bukti
Diketahui :

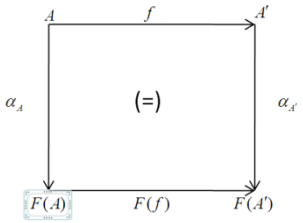
$$((1_D \times f)^I \circ (\gamma \otimes \delta))(s) = (1_D \times f)^I \circ ((\gamma \otimes \delta)(s))$$
$$= ((1_D \times f) \circ (\gamma \otimes \delta))(s) \text{ untuk setiap } state\ s \text{ dan } input\ e$$
f adalah homomorfisma automata
$$\Leftrightarrow \big(\gamma(s, e), (f \circ \delta)(s, e)\big) = ((\gamma'(f(s), e), \delta'(f(s), e))$$
$$\Leftrightarrow ((1_D \times f) \circ (\gamma \otimes \delta))(s, e) = (\gamma' \otimes \delta')(f(s), e)$$
$$\Leftrightarrow (1_D \times f)((\gamma \otimes \delta)(s)(e) = ((\gamma' \otimes \delta') \circ f)(s)(e)$$
$$\Leftrightarrow ((1_D \times f)^I \circ (\gamma \otimes \delta))(s)(e) = ((\gamma' \otimes \delta') \circ f)(s)(e) \quad \square$$

Diantara struktur homomorfisma automata dengan homomorfisma suatu kotak hitam ada hubungannya. Suatu kotak hitam adalah koaljabar dengan suatu pemetaan $\alpha_S : S \to F(S)$, dimana F adalah pemetaan

suatu nilai fungsi (set-value) dengan $F(X) = D \times X$ untuk setiap himpunan X. Sedangkan suatu homomorfisma automata dipandang sebagai suatu koaljabar dengan pemetaan $\alpha_S : S \to T(S)$, dimana pemetaan nilai fungsi T diartikan sebagai $T(X) = (D \times X)^I$ untuk setiap *state* X. Kedua pemetaan F dan T dapat dikenakan pemetaan berikut ini:

Misalkan $f : S \to S'$ maka $F(f) = 1_D \times f : D \times S \to D \times S'; T(f) = (1_D \times f)^I : (D \times S)^I \to (D \times S')^I$, maka $(S; \alpha_S)$ dan $(S'; \alpha_S)$ adalah representasi dari kotak hitam dan automata.

Suatu pemetaan $f : S \to S'$ adalah homomorfisma jika dan hanya jika $F(f) \circ \alpha_S = \alpha_{S'} \circ f$. Dapat dilihat dari diagram dibawah ini yang komutatif.



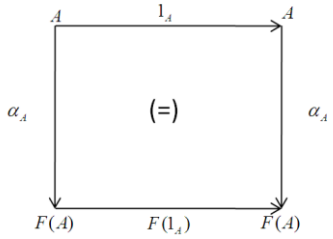
Gambar 5. Homomorfisma F-koaljabar

Hal ini menunjukkan bahwa jika homomorfisma adalah pemetaan yang mengawetkan suatu struktur maka minimal ada dua sifat yang harus dipertahankan :

(i) Pemetaan identitas adalah suatu homomorfisma
(ii) komposisi dua homomorfisma adalah homomorfisma juga

Dalam koaljabar, jika $f : A \to B$ maka selalu terdapat $F(f) : F(A) \to F(B)$. Berdasarkan sifat tersebut maka akan dilihat apakah dapat diperoleh sifat yang lain jika dua sifat homomorfisma di atas juga dipenuhi.

Untuk setiap $(A; \alpha_A)$, suatu pemetaan 1_A adalah homomorfisma jika diagram berikut ini komutatif.

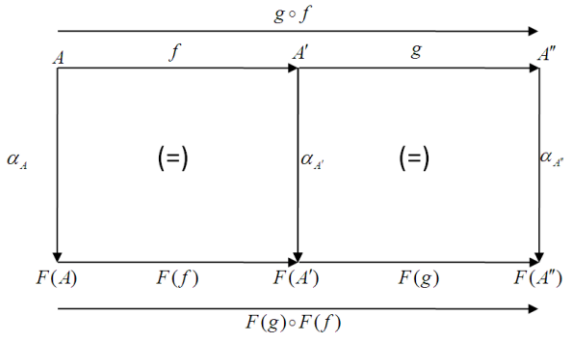


Gambar 6. Homomorfisma Identitas

Dari diagram terlihat bahwa $\alpha_A \circ 1_A = F(1_A) \circ \alpha_A$, hal ini menunjukkan bahwa F mengawetkan suatu pemetaan identitas

$$1_A,\ F(1_A) = 1_{F(A)}$$

Misalkan $(A; \alpha_A)$, $(A'; \alpha_{A'})$ dan $(A''; \alpha_{A''})$ adalah F-koaljabar dan misalkan $f : (A; \alpha_A) \to (A'; \alpha_{A'})$ dan $g : (A'; \alpha_{A'}) \to (A''; \alpha_{A''})$ adalah homomorfisma. Jika f dan g adalah homomorfisma maka dua persegi yang kecil dari diagram berikut ini akan komutatif



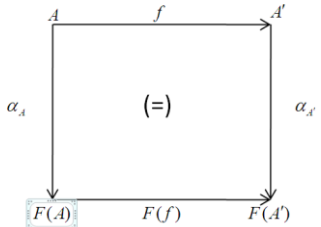
Gambar 7. Komposisi homomorfisma

Suatu pemetaan $g \circ f$ adalah homomorfisma jika persegi yang paling besar dari diagram diatas komutatif sedemikian hingga $\alpha_A \circ (g \circ f) = F(g \circ f) \circ \alpha_A$ dengan anggapan bahwa $F(g \circ f) = F(g) \circ F(f)$

Definisi 15:
Fungtor pada himpunan adalah suatu operasi F pada himpunan tersebut dan pemetaan yang memenuhi :

(i) Jika A adalah himpuan maka $F(A)$ juga himpunan.
(ii) Jika f adalah pemetaan maka $F(f)$ juga pemetaan dengan sifat :
  a. Jika $f : A \to B$ maka $F(f) : F(A) \to F(B)$
  b. $F(1_A) = 1_{F(A)}$
  c. $F(f \circ g) = F(f) \circ F(g)$ adalah komposisi dimana $f : A \to B$ dan $g : B \to C$.

Definisi 16:
Misalkan $F : Set \to Set$ adalah functor dan $(A; \alpha_A)$ dan $(A'; \alpha_{A'})$ adalah F-koaljabar. Suatu pemetaan $f : A \to B$ adalah homomorfisma koaljabar dari $(A; \alpha_A)$ ke $(A'; \alpha_{A'})$ jika $F(f) \circ \alpha_A = \alpha_{A'} \circ f$ dan diagram berikut ini komutatif.



Gambar 8. Homomorfisma koaljabar

Proposisi 6:
Misalkan $F : Set \to Set$ *adalah functor maka untuk setiap koaljabar* $(A;\alpha_A)$, $(A';\alpha_{A'})$ $(A'';\alpha_{A''})$ *berlaku :*

(i) $1_A : (A;\alpha_A) \to (A;\alpha_A)$ *adalah homomorfisma koaljabar*

(ii) *Jika* $f : (A;\alpha_A) \to (A';\alpha_{A'})$ *dan* $g : (A';\alpha_{A'}) \to (A'';\alpha_{A''})$ *adalah homomorfisma koaljabar maka* $g \circ f : (A;\alpha_A) \to (A'';\alpha_{A''})$

KESIMPULAN

1. Koaljabar dalam sistem *state-based* merupakan kombinasi beberapa pemetaan dari sistem *state-based* menjadi suatu pemetaan.
2. Kotak hitam dalam sistem *state-based* menggunakan prinsip koaljabar dengan menggabungkan pemetaan $\alpha : D \to E$, $\beta : D \to F$ menjadi $\alpha \otimes \beta : D \to E \times F$ yang merupakan hasil kali tensor dari α dan β sehingga dapat ditulis $(S, \alpha \otimes \beta)$.
3. Automata dalam sistem *state-based* menggunakan prinsip koaljabar dengan menggabungkan pemetaan $\alpha : S \times I \to S$ dan $\gamma : S \times I \to O$ menjadi suatu pemetaan yaitu $\alpha_S : S \to O \times S^I$ dan $\alpha_S : S \to (O \times S)^I$.
4. Struktur Kripke dalam sistem *state-based* menggunakan prinsip koaljabar dengan menggabungkan pemetaan $next : S \to P(S)$ dan $prop : S \to P(\Phi)$ sehingga struktur Kripke $(S; R; \upsilon)$ dapat dipandang sebagai $(S; next \otimes prop)$ dimana $next \otimes prop : S \to P(S) \times P(\Phi)$

DAFTAR PUSTAKA

[1] Denecke K., Wismath, S.L.,2009, *Universal Algebra and Coalgebra*, World Scientific. New York

[2] Gumm, H.P., 2009, *Universal Coalgebras and Their Logics*, The Arabian Journal for Sciene and Engineering (AJSE), Volume I, p. 105-130

[3] Jacobs, B, 2005, *Introduction to Coalgebra Towards Mathematics of States and Observations*, Institute for Computing and Information Sciences, Radbound University Nijmegen, Netherlands

[4] Rutten, J., 2000, *Universal Coalgebra A Theory of System Theoritical Computing Science*, p 249, Elsevier

[5] Kupke, C., Kurz, A., and Pattinson, D., 2004, *Algebraic Semantic for Coalgebraic Model Logic*, Electronic Notes in Theoritical Computer Science, p 106, Elsevier

[6] Hasuo, L., Jacobs, B., and Sokolova, A., 2007, *Generic Trace Semantic via Coinduction*, Logical Methods in Computer Science 3, Issue A, pp 1-36

[7] Hansen, H.,H., and Ruten, J., 2014, *Stream and Coalgebra*, Radbound University Nijmegen & CWI Amsterdam